

# Semantic Web Technologies 1

Sebastian Rudolph und Elena Simperl

Wintersemester 2011/12

<http://semantic-web-grundlagen.de>

## Lösung der Übung 2: Logik und RDF-Semantik

### Lösung 2.1

(a)  $(p \vee \neg p)$ : **allgemeingültig**

$I(p)$	$I(\neg p)$	$I(p \vee \neg p)$
$t$	$f$	$t$
$f$	$t$	$t$

(b)  $((p \vee q) \rightarrow (\neg p \vee \neg q))$ : **erfüllbar & widerlegbar**

$I(p)$	$I(q)$	$I(\neg p)$	$I(\neg q)$	$I(p \vee q)$	$I(\neg p \vee \neg q)$	$I((p \vee q) \rightarrow (\neg p \vee \neg q))$
$t$	$t$	$f$	$f$	$t$	$f$	$f$
$t$	$f$	$f$	$t$	$t$	$t$	$t$
$f$	$t$	$t$	$f$	$t$	$t$	$t$
$f$	$f$	$t$	$t$	$f$	$t$	$t$

(c)  $\neg((p \rightarrow q) \leftrightarrow (\neg p \vee q))$ : **unerfüllbar**

$I(p)$	$I(q)$	$I(\neg p)$	$I(p \rightarrow q)$	$I(\neg p \vee q)$	$I((p \rightarrow q) \leftrightarrow (\neg p \vee q))$	$I(\neg((p \rightarrow q) \leftrightarrow (\neg p \vee q)))$
$t$	$t$	$f$	$t$	$t$	$t$	$f$
$t$	$f$	$f$	$f$	$f$	$t$	$f$
$f$	$t$	$t$	$t$	$t$	$t$	$f$
$f$	$f$	$t$	$t$	$t$	$t$	$f$

(d)  $((p \rightarrow q) \rightarrow p) \rightarrow p$ : **allgemeingültig**

$I(p)$	$I(q)$	$I((p \rightarrow q) \rightarrow p)$	$I(((p \rightarrow q) \rightarrow p) \rightarrow p)$
$t$	$t$	$t$	$t$
$t$	$f$	$f$	$t$
$f$	$t$	$t$	$t$
$f$	$f$	$t$	$t$

(e)  $((p \wedge q) \rightarrow r) \leftrightarrow (p \rightarrow (q \rightarrow r))$ : **allgemeingültig**

$I(p)$	$I(q)$	$I(r)$	$I((p \wedge q))$	$I(((p \wedge q) \rightarrow r))$
$t$	$t$	$t$	$t$	$t$
$t$	$t$	$f$	$t$	$f$
$t$	$f$	$t$	$f$	$t$
$t$	$f$	$f$	$f$	$t$
$f$	$t$	$t$	$f$	$t$
$f$	$t$	$f$	$f$	$t$
$f$	$f$	$t$	$f$	$t$
$f$	$f$	$f$	$f$	$t$

$I(q \rightarrow r)$	$I((p \rightarrow (q \rightarrow r)))$	$I(((p \wedge q) \rightarrow r) \leftrightarrow (p \rightarrow (q \rightarrow r)))$
$t$	$t$	$t$
$f$	$f$	$t$
$t$	$t$	$t$
$t$	$t$	$t$
$t$	$t$	$t$
$f$	$t$	$t$
$t$	$t$	$t$
$t$	$t$	$t$

(f)  $((p \wedge \neg p) \rightarrow q)$ : **allgemeingültig**

$I(p)$	$I(\neg p)$	$I((p \wedge \neg p))$	$I(((p \wedge \neg p) \rightarrow q))$	
$t$	$t$	$f$	$f$	$t$
$t$	$f$	$f$	$f$	$t$
$f$	$t$	$t$	$f$	$t$
$f$	$f$	$t$	$f$	$t$

## Lösung 2.2

*Theorie*: Eine *Theorie* ist eine Menge von Sätzen.

*logische Konsequenz*:  $\phi$  ist logische Konsequenz von  $\mathcal{T}$  wenn  $\mathcal{T} \models \phi$ .

*Äquivalenz*:  $\phi \equiv \psi$  wenn  $\{\phi\} \models \psi$  und  $\{\psi\} \models \phi$ .

Für beliebige Theorien  $\mathcal{T}$  und  $\mathcal{S}$  gilt:

(a) Ist eine Formel  $F$  allgemeingültig, dann gilt  $\mathcal{T} \models F$ , d.h. aus jeder Theorie folgen zumindest alle Tautologien.

♣ **wahr**: Für beliebige Interpretation  $I$ , wenn  $I \models \mathcal{T}$ , dann  $I \models F$

(b) Je größer eine logische Theorie ist, desto mehr Modelle hat sie. Das heißt, wenn  $\mathcal{T} \subseteq \mathcal{S}$ , dann ist jedes Modell von  $\mathcal{T}$  auch ein Modell von  $\mathcal{S}$ .

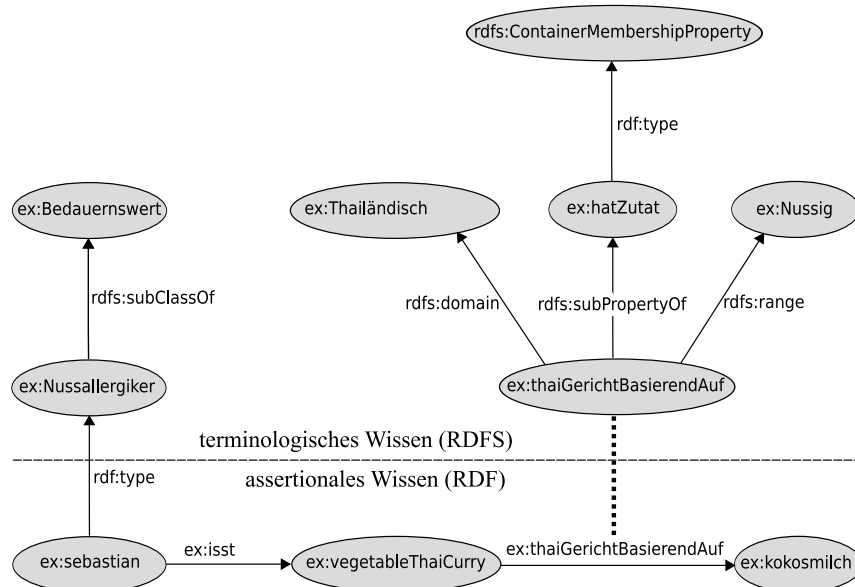
♠ **falsch**: Gegenbeispiel:  $\mathcal{T} = \{p\}$ ,  $\mathcal{S} = \{p, \neg p\}$ .  $\mathcal{T}$  ist erfüllbar und  $\mathcal{S}$  ist unerfüllbar.

- (c) Je größer eine Theorie ist, desto mehr logische Konsequenzen hat sie. Das heißt, wenn  $\mathcal{T} \subseteq \mathcal{S}$ , dann ist jede logische Konsequenz aus  $\mathcal{T}$  auch eine Konsequenz aus  $\mathcal{S}$ .  
 ♣ **wahr**: "monotonic Logik."
- (d) Ist  $\neg F \in \mathcal{T}$ , dann kann  $\mathcal{T} \models F$  niemals gelten (wobei  $F$  eine beliebige Formel ist).  
 ♣ **falsch**: im Allgemeinen.  
 ♣ **wahr**: Für beliebige Modell  $\mathcal{I}$  einer **consistent** Theorie  $\mathcal{T}$  und  $F' \in \mathcal{T}$ ,  $\mathcal{I} \models F'$ , insbesondere,  $\neg F \in \mathcal{T} \Rightarrow \mathcal{I} \models \neg F$  oder  $\mathcal{I} \not\models F$  d.h.  $\mathcal{T} \models F$  kann niemals gelten.
- (e) Sind zwei Theorien unterschiedlich ( $\mathcal{T} \neq \mathcal{S}$ ), dann unterscheiden sie sich auch in wenigstens einer logischen Konsequenz (zum Beispiel, indem es eine Formel  $F$  gibt, so dass  $\mathcal{T} \models F$  aber  $\mathcal{S} \not\models F$ ).  
 ♣ **falsch**: Für zwei Äquivalenz Theorie  $\mathcal{T}$  und  $\mathcal{S}$  mit  $\mathcal{T} \neq \mathcal{S}$ , wenn  $\mathcal{T} \models F$  (für beliebige Formel  $F$ ), dann jedes Modell von  $\mathcal{T}$  ist ein Modell von  $\mathcal{S}$ . Aber jedes Modell von  $\mathcal{T}$  ist auch ein Modell von  $\mathcal{S}$  ( $\mathcal{T} \equiv \mathcal{S}$ ) d.h.  $\mathcal{S} \models F$ , z.B.  $\{p, q\}$  und  $\{p \wedge q\}$ ,  $\{p, q\} \neq \{(p \wedge q)\}$  aber  $\{p, q\} \models (p \wedge q)$  und  $\{(p \wedge q)\} \models (p \wedge q)$ .

### Lösung 2.3

$\mathcal{I}$  ist eine Interpretation des Modells der Graph.

- $IR = \{a\}$
- $IP = \{a\}$
- $I_{EXT}(a) = \{\langle a, a \rangle\}$
- $I_S(x) = a$ ,  $x$  ist beliebige Resource in Graph.
- $LV = I_L = \emptyset$



1.  $\mathcal{I}$  ist eine (einfache) Interpretation. ✓
2.  $\mathcal{I}$  ist eine RDF-interpretation: ✓
3.  $\mathcal{I}$  ist eine RDFS-interpretation: ✓

#### Lösung 2.4

- ein Tripel, welches einfach folgt:

ex:vegetableThaiCurry	ex:thaiGerichtBasierendAuf	ex:kokosmilch
⇓ (Regel se1)		
ex:vegetableThaiCurry	ex:thaiGerichtBasierendAuf	_ : id1

- ein Tripel, welches RDF-folgt, aber nicht einfach folgt:

ex:vegetableThaiCurry	ex:thaiGerichtBasierendAuf	ex:kokosmilch
⇓ (Regel rdf1)		
ex:thaiGerichtBasierendAuf	rdf:type	rdf:Property

- ein Tripel, welches RDFS-folgt, aber nicht einfach folgt:

ex:vegetableThaiCurry	ex:thaiGerichtBasierendAuf	ex:kokosmilch
<b>Und</b>		
ex:thaiGerichtBasierendAuf	rdfs:domain	ex:Thailändisch
⇓ (Regel rdfs2)		
ex:vegetableThaiCurry	rdf:type	ex:Thailändisch

#### Lösung 2.5

Nicht möglich in RDFS.

#### Lösung 2.6

(a) `rdfs:Resource rdf:type rdfs:Class .`

$\frac{\text{rdfs:domain rdfs:range rdfs:Class .} \quad \text{rdf:type rdfs:domain rdfs:Resource .}}{\text{rdfs:Resouce rdf:type rdfs:Class .}} \text{ rdfs3}$
--

(b) `rdfs:Class rdf:type rdfs:Class .`

$$\frac{\text{rdfs:range rdfs:range rdfs:Class .} \quad \text{rdfs:range rdfs:range rdfs:Class .}}{\text{rdfs:Class rdf:type rdfs:Class .}} \text{ rdfs3}$$

(c) `rdfs:Literal rdf:type rdfs:Class .`

$$\frac{\text{rdfs:range rdfs:range rdfs:Class .} \quad \text{rdfs:comment rdfs:range rdfs:Literal .}}{\text{rdfs:Literal rdf:type rdfs:Class .}} \text{ rdfs3}$$

(d) `rdf:XMLLiteral rdf:type rdfs:Class .`

$$\frac{\text{rdfs:subClassOf rdfs:domain rdfs:Class .} \quad \text{rdf:XMLLiteral rdfs:subClassOf rdfs:Literal .}}{\text{rdf:XMLLiteral rdf:type rdfs:Class .}} \text{ rdfs2}$$

(e) `rdfs:Datatype rdf:type rdfs:Class .`

$$\frac{\text{rdf:type rdfs:range rdfs:Class .} \quad \text{rdfs:XMLLiteral rdfs:type rdfs:Datatype .}}{\text{rdfs:Datatype rdf:type rdfs:Class .}} \text{ rdfs3}$$

(f) `rdf:Seq rdf:type rdfs:Class .`

$$\frac{\text{rdfs:subClassOf rdfs:domain rdfs:Class .} \quad \text{rdf:Seq rdfs:subClassOf rdfs:Container .}}{\text{rdf:Seq rdf:type rdfs:Class .}} \text{ rdfs2}$$

(g) `rdf:Bag rdf:type rdfs:Class .`

$$\frac{\text{rdfs:subClassOf rdfs:domain rdfs:Class .} \quad \text{rdf:Bag rdfs:subClassOf rdfs:Container .}}{\text{rdf:Bag rdf:type rdfs:Class .}} \text{ rdfs2}$$

(h) `rdf:Alt rdf:type rdfs:Class .`

$$\frac{\text{rdfs:subClassOf rdfs:domain rdfs:Class .} \quad \text{rdf:Alt rdfs:subClassOf rdfs:Container .}}{\text{rdf:Alt rdf:type rdfs:Class .}} \text{ rdfs2}$$

(i) `rdfs:Container rdf:type rdfs:Class .`

$$\frac{\text{rdfs:subClassOf rdfs:range rdfs:Class .} \quad \text{rdf:Alt rdfs:subClassOf rdfs:Container .}}{\text{rdfs:Container rdf:type rdfs:Class .}} \text{ rdfs3}$$

(j) `rdf:List rdf:type rdfs:Class .`

$$\frac{\text{rdfs:domain rdfs:range rdfs:Class .} \quad \text{rdf:first rdfs:domain rdf:List .}}{\text{rdf:List rdf:type rdfs:Class .}} \text{ rdfs3}$$

- (k) `rdfs:ContainerMembershipProperty rdf:type rdfs:Class .`  

$$\frac{\text{rdfs:subClassOf rdfs:domain rdfs:Class .} \quad \text{rdf:ContainerMembershipProperty rdfs:subClassOf rdfs:Property .}}{\text{rdfs:ContainerMembershipProperty rdf:type rdfs:Class .}} \text{ rdfs2}$$
- (l) `rdf:Property rdf:type rdfs:Class .`  

$$\frac{\text{rdfs:range rdfs:range rdfs:Class .} \quad \text{rdf:subPropertOf rdfs:range rdf:Property .}}{\text{rdf:Property rdf:type rdfs:Class .}} \text{ rdfs3}$$
- (m) `rdf:Statement rdf:type rdfs:Class .`  

$$\frac{\text{rdfs:domain rdfs:range rdfs:Class .} \quad \text{rdf:Subject rdfs:domain rdf:Statement .}}{\text{rdf:Statement rdf:type rdfs:Class .}} \text{ rdfs3}$$
- (n) `rdfs:domain rdf:type rdf:Property .`  

$$\frac{\text{rdfs:range rdfs:domain rdf:Property .}}{\text{rdfs:domain rdf:type rdf:Property .}} \text{ rdf1}$$
- (o) `rdfs:range rdf:type rdf:Property .`  

$$\frac{\text{rdfs:subPropertyOf rdfs:range rdf:Property .}}{\text{rdfs:range rdf:type rdf:Property .}} \text{ rdf1}$$
- (p) `rdfs:subPropertyOf rdf:type rdf:Property .`  

$$\frac{\text{rdfs:isDefinedBy rdfs:subPropertyOf rdfs:SeeAlso .}}{\text{rdfs:subPropertyOf rdf:type rdf:Property .}} \text{ rdf1}$$
- (q) `rdfs:subClassOf rdf:type rdf:Property .`  

$$\frac{\text{rdf:Alt rdfs:subClassOf rdfs:Container .}}{\text{rdfs:subClassOf rdf:type rdf:Property .}} \text{ rdf1}$$
- (r) `rdfs:member rdf:type rdf:Property .`  

$$\frac{\text{rdfs:range rdfs:domain rdf:Property .} \quad \text{rdfs:member rdfs:range rdfs:Resource}}{\text{rdfs:member rdf:type rdf:Property .}} \text{ rdfs2}$$
- (s) `rdfs:seeAlso rdf:type rdf:Property .`  

$$\frac{\text{rdfs:range rdfs:domain rdf:Property .} \quad \text{rdfs:seeAlso rdfs:range rdfs:Resource .}}{\text{rdfs:seeAlso rdf:type rdf:Property .}} \text{ rdfs2}$$

(t) `rdfs:isDefinedBy rdf:type rdf:Property .`

`rdfs:range rdfs:domain rdf:Property .    rdfs:isDefinedBy rdfs:range rdfs:Resource .` `rdfs2`  
`rdfs:isDefinedBy rdf:type rdf:Property .`

(u) `rdfs:comment rdf:type rdf:Property .`

`rdfs:range rdfs:domain rdf:Property .    rdfs:comment rdfs:range rdfs:Literal .` `rdfs2`  
`rdfs:comment rdf:type rdf:Property .`

(v) `rdfs:label rdf:type rdf:Property .`

`rdfs:range rdfs:domain rdf:Property .    rdfs:label rdfs:range rdfs:Literal .` `rdfs2`  
`rdfs:label rdf:type rdf:Property .`